Technical report

Bard VADER: Bard as a **V**irtual **A**ssistant **D**evice for **E**xpert Art **R**ecognition.

1. Introduction

Nowadays, generalized access to the internet enables artists to share their work with the world. This results in a rising interest in art and culture, and increases people's curiosity about the greatest art masters in history. Consequently, museums and art galleries have seen an increment in their attendance. However, these places have a limited number of personal guides, and the generally provided information through booklets or audioguides is sometimes not enough to answer customers' inquiries. Thus, there is a need for more flexible, readily-available systems that are able to provide customized information and adapt to the requirements of each individual. In this context, we propose Bard VADER, an AI-based virtual museum guide capable of detecting a piece of art using computer vision methods and retrieving information about the work using a state-of-the-art Large Language Model (LLM) like Bard.Our system workflow is divided into different stages. First, the art piece is detected using a computer vision algorithm that matches the features extracted from the image with a database of the pieces showcased at the museum/gallery. Then, when the work has been identified, we ask the LLM about information from the detected piece, such as its author, epoch, art style, or historical context. Finally, the user is able to interact with the LLM and ask about any specific topic that he/she is interested in, providing a personalized experience for each customer.This system can be implemented either in an embedded system like a Khadas Edge2 platform, allowing for each customer to have all the information in their own hands, or embodied in a robot platform that serves as a guide and is able to provide a tour through all the place. The advantage of our proposal is that it provides a neverending source of information that adapts to each individuals interests, making the experience of visiting a museum or an art gallery more interactive, engaging, and satisfying than ever before. In addition, our solution is relatively low cost, since it only requires a camera, a processing unit, and connection to an online server to retrieve the required information from the LLM.

2. Methodology

2.1. Data acquisition

Two methods have been implemented to acquire the image in real-time, as our application is designed to be used in a cultural context such as a museum. Therefore, the image of the artwork

for which the user seeks information can be captured either by a webcam or from their own mobile phone through an application.

If you choose to use the webcam, OpenCV already includes a method for camera access. If opting for a mobile phone, you'll need to download the following application. This app enables the retrieval of frames captured by the device's camera through the IP address (Figure 1).

- IP Webcam (Android) : https://play.google.com/store/apps/details?id=com.pas.webcam&hl=es&gl=US&pli=1

Figura 1: IP Webcam

Figure 2: Photo taken by the smartphone camera.

2.2. Pre-processing image

Once we have obtained the image (Figure 2), it is necessary to preprocess it. In this stage, the image is scaled to accelerate computational speed. Subsequently, the contours of the frame are extracted, which represent the significant part of the image. The subsequent stages of processing will then work with this newly cropped image (Figure 3).

Figure 3: Preprocessed image.

2.3. Feature Extraction

As mentioned before, once you have an image of the artwork you want to learn more about, you can use a matching process to identify it in a database. This will tell you the name of the artwork and its artist.

ORB is the chosen feature detector for extracting features from both the user-captured image and the database images. ORB (Oriented FAST and Rotated BRIEF) is a feature detection and description algorithm that is widely used in computer vision applications. It is a fast and efficient alternative to SIFT (Scale-Invariant Feature Transform) and SURF (Speeded-Up Robust Features), and it has been shown to be effective for a variety of tasks, including object recognition, image registration, and 3D reconstruction.

ORB, a feature detector and descriptor that stands out for its efficiency and robustness, relies on FAST, a keypoint detector known for its speed and ability to detect keypoints in low-contrast images, and BRIEF, a binary descriptor that remains unaffected by illumination changes and minor rotations.

Additionally, it extends the functionality of FAST and BRIEF by introducing a collection of enhancements, including multi-scale feature detection, enabling robust keypoint detection across varying image scales; oriented BRIEF, providing keypoint orientation estimation for rotation-invariant descriptors; and Hamming distance matching, employing an efficient method for binary descriptor matching.

To perform the matching between the artwork and the database, the first step is to detect keypoints in the images. ORB uses a FAST algorithm to identify keypoints in the images. Once the keypoints have been detected, they need to be described. ORB uses a BRIEF algorithm to describe the keypoints. (Figure 4)

Figure 4: Keypoints extracted by ORB.

The keypoints matching algorithm compares the descriptors of the keypoints in the image to the descriptors of the keypoints in the database. For each keypoint in the image, the algorithm finds the keypoint in the database that has the closest descriptor. The result of this comparison is a correspondence matrix, where each row represents a pair of corresponding keypoints (Figure 5).

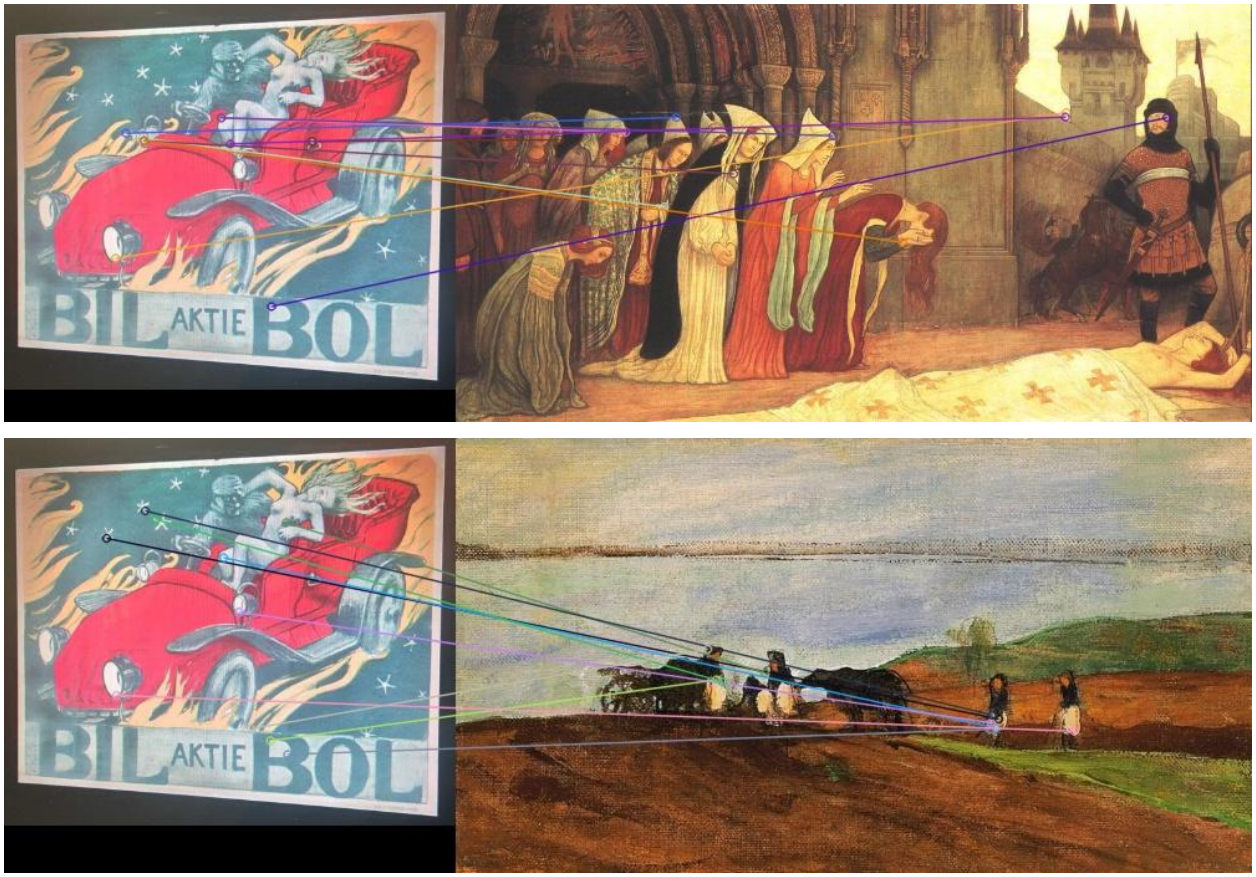

Figure 5: Examples of matches

Once the keypoints matching has been performed, the quality of the matching can be evaluated using a matching metric, such as the Hamming distance. The total distance of the 10 best matches for each of the dataset images is calculated. The image that minimizes this distance will be the artwork we are looking for (Figure 6).

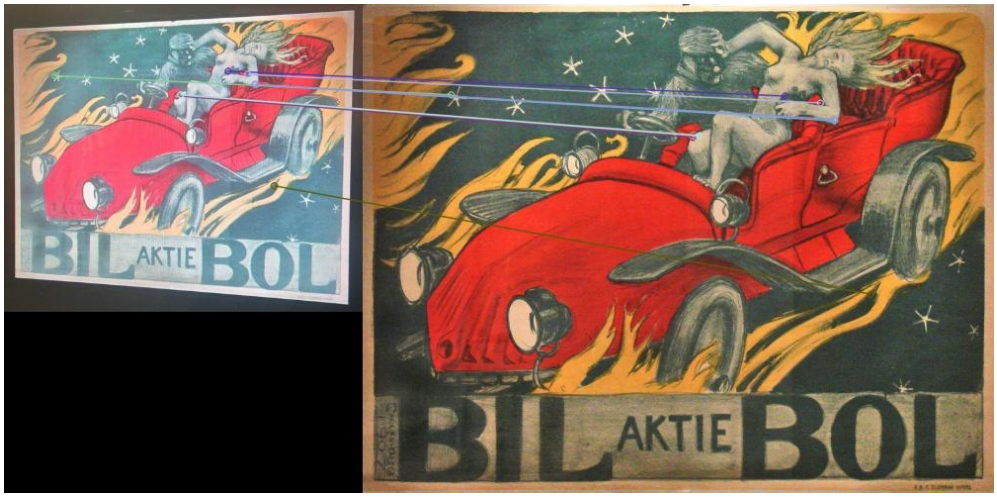

Figure 6: Best match

Once the artwork is known, we can extract the title and the author, which will be used as input to the LLM.

2.4. BARD

After processing the captured image and detecting the most likely match in our database of artwork, we can now connect to an LLM API and start a conversation to ask for information about the recognized artwork.

In our case, we've chosen Google Bard as the LLM to retrieve information from. This decision has been made due to the convenience of the provided Python API, and also because it can be used freely. In addition, Bard is also connected to the internet, so it can provide a more updated information than other options like LLaMa. Nonetheless, for a more professional, real-world use, other options could be considered. These options include OpenAI's ChatGPT, or even a fine-tuned GPT model specialized in Art.

To set up the connection to Bard, there are a series of configuration steps that need to be performed before running the application. These steps have been detailed in the README file provided with the code repository for ease of use, and basically consist of retrieving personal tokens related to the user's Google Bard account.

Back to our application execution, at this point we'll have a string containing the name of the detected artwork and its author. This information we'll be added to a pre-defined initial prompt to the LLM, resulting in something like: *"Provide information about the artwork X?"*, with X being the recognized artwork. We can add a customized pre-prompt as well in order to modulate the questions being asked. For example, in the default version of the code we've added a pre-prompt to limit Bard's answer to no more than 100 words for briefness.

Using the API, we retrieve Bard's provided information about the artwork we specified and show it to the user. In our prototype, this information is shown as text in the terminal executing the application. However, we could easily add a voice synthetizer to also hear the answer. Moreover, in a real-world use case, the answer would be provided directly on the app being ran on the Smartphone or the Khadas Edge2 platform, directly on the user's hands.

After this first answer, the application enters conversation mode. The user is now able to ask follow-up questions regarding the recognized artwork, such as more details about the artwork, other pieces of work from the same author, other similar artworks in the same historical context, etc. It's up to the user how many questions to ask and when to stop the current conversation, thus enabling a totally customized experience.

These questions are asked directly with natural language, and we then translate the audio into text and submit the question to the LLM via Bard's API. We decided to use voice commands because it's the best way to resemble having a personal guide and interacting with it.

Whenever the user wants to stop the current conversatio, it only needs to say "STOP" when the application asks for the next follow-up question, and then the loop and be ready to start again from the beginning.

3. Results

A demonstration video of the execution of our application can be watched at this YouTube link. It can be seen how we use the camera of a mobile phone to get the image of a piece of artwork. Then, we recognize the artwork in our database and ask Bard for basic information about it. Then, we ask a followup question to get more information about the author, which Bard retrieves correctly again. We only show one followup question for briefness, but as mentioned before, one can ask as many questions as required.

4. Conclusion

We have presented Bard Vader: an AI virtual museum guide. We propose an application that allows users to live a customized experience when visiting museums, art galleries, or other kind of cultural locations. The user can take a picture of a piece of artwork of their interest, and the app will perform feature extraction and matching using computer vision techniques to the pieces of artwork present in a pre-defined database, thus recognizing the artwork chosen by the user. Then, it will connect to an online LLM, in our case Google Bard, and retrieve information about the recognized artwork. After this first round of information, the user can ask as many questions as they want via natural speech in order to satisfy all their curiosity about the artwork, getting a fully personalized experience.

We have implemented a first prototype of Bard Vader and test it under controlled laboratory conditions. The results shown in the previous section confirm that with our application one can get a customized experience similar to have a personal guide right in your pocket. However, there are some limitations that need to be addressed. In its current state, the interaction with Bard needs to refresh the tokens every few minutes, which makes continued use very difficult. In addition, more robust and sophisticated image recognition techniques can be explored, such as learning-based methods.

Future work should seek to switch to a more stable version of an LLM API, enabling a continued use of the application. Furthermore, developing everything as a mobile application or inside an embedded system will improve user experience and get closer to a possible real-world application. An ideal scenario would be one where all the paintings of an specific art gallery are included in the database, and portable devices are given to the visitors with the application running inside, allowing them to interact with a tailored version of an LLM specialized in history of art.